Attorney Docket No.: 7781.0156-00
SAP Ref. No. 2003P00493 US

UNITED STATES PATENT APPLICATION

FOR

AUTOMATIC TRANSLATION CODE GENERATION

BY

JUAN CARLOS MARTINEZ

RALF PHILIPP

FRANK SOMMERLADE

AUTOMATIC TRANSLATION CODE GENERATION

Field of the Invention

[001] The present invention relates to computer-implemented methods and systems for automatically configuring a translation code for use in translating data stored in a server into a data format required by a client.

Background

[002] In large corporations and institutions, computer networks today are generally comprised of many different data management systems and thus provide a heterogeneous landscape. Moreover, over time, such data management systems frequently become more and more heterogeneous, as new systems are integrated into the landscape and existing systems are updated.

[003] Such data management systems may store and provide data in any number of different data formats. The use of standard messaging technologies, like extended markup language (XML), facilitates the providing and exchanging data between different systems with reduced difficulty; however, not all data management systems use standardized formats and languages. Therefore, many times when a client requests data from a server, the relevant data may first have to be converted from one format to another before it is transmitted from the server to the client.

[004] Sometimes, manual hard coding of the necessary conversion routines is done for transmitting converted data from a server to a client. Hard coding of the conversion routines, however, is costly, inflexible, and labour intensive. As a result, rather than manually change a hard-coded conversion routine, it is common practice to

provide more data to the client than actually necessary and relevant for the particular client. This practice consumes bandwidth and produces traffic that is not necessary.

[005] Automatic translation code generation would provide the data requested by the client in an appropriate data format and allow transfer of only the relevant data ma, thus saving bandwidth. One example of a known code generator is described in Christian Georgescu, "Code Generation Templates Using XML and XSL", C/C^{++} Users Journal, January 2002, p. 6-19. In Georgescu, an implementation of XML or XSL for Java and C++ code generation is discussed. These code generators allow generation of C^{++} or Java computer code from a data set of a different data format. A meta-model of the provided data provides a structure for an information model that comprises data in XML-format. Using the meta-model, a design logic may be coded in XSL for generating implementation logic that allows a code generator to code the information model into a different data format without manual labour.

[010] However, even with known code generators, it is not possible to react automatically to changes within a data format required by a client. The known code generators alone do not provide automatic adaptation to changes in a data object model, e.g a data format, on the side of the client.

SUMMARY

[010] The present invention provides methods and systems for automatic translation code generation. In certain embodiments consistent with the present invention, the client transmits a data object message comprising the current data model of the client to the server. The server automatically generates a new translation code based on the current data model of the client. The newly generated translation code

may then be used to convert the data in the server into a data format required by the client.

[011] Certain embodiments consistent with the present invention provide the data requested by the client in an appropriate data format. When only relevant data is transferred, unnecessary data need not be transmitted and bandwidth may be saved. Moreover, by automatically generating translation codes, manual re-coding is unnecessary and the time needed to react to changes may be minimized. Furthermore, by avoiding manual re-coding, coding errors and the number of personnel needed to implement changes may be reduced. Different and new clients may be quickly added or removed from the network. Clients only need to be able to deliver data object definition messages to the client. The data object definition messages comprise information about the data format required by the client. The knowledge of this data format allows the server to adapt a translation code to a certain client.

BRIEF DESCRIPTION OF THE DRAWINGS

[010] The accompanying drawings, which are incorporated in and constitute a part of this specification, illustrate various embodiments of the invention and, together with the description, serve to explain the principles of the invention. In the drawings:

[011] FIG. 1illustrates an exemplary computer system used to implement one or more embodiments of the present invention;

[012] FIG. 2 is a diagram illustrating translation code generation after a change in the object model, according to one or more embodiments of the present invention; and

[013] FIG. 3 is a diagram illustrating translation code generation after receiving data with a new data object definition, according to one or more embodiments of the present invention.

DETAILED DESCRIPTION

[010] Methods and systems consistent with the principles of the present invention enable a server to automatically generate a new translation code based on the current data model of the client. The newly generated translation code may then be used to convert the data in the server into a data format required by the client.

[011] FIG. 1 illustrates a simplified block diagram of an exemplary computer system 999 for implementing the principles of the present invention. As shown in FIG. 1, computer system 999 may have a plurality of computers 900, 901, 902 (or even more). Computer 900 can communicate with computers 901 and 902 over network 990. Computer 900 has processor 910, memory 920, bus 930, and, optionally, input device 940 and output device 950 (I/O devices, user interface 960). As illustrated, the invention is implemented by computer program product 100 (CPP), carrier 970 and signal 980.

[012] With respect to computer 900, computer 901/902 is sometimes referred to as a "remote computer." Computer 901/902 may be, for example, a server, a peer device or other common network node, and may have many or all of the elements described relative to computer 900.

[013] Computer 900 may be, for example, a conventional personal computer (PC), a desktop device or a hand-held device, a multiprocessor computer, a pen computer, a microprocessor-based or programmable consumer electronics device, a

minicomputer, a mainframe computer, a personal mobile computing device, a mobile phone, a portable or stationary personal computer, a palmtop computer or the like.

[014] Processor 910 may be, for example, a central processing unit (CPU), a micro-controller unit (MCU), a digital signal processor (DSP), or the like.

[015] Memory 920 may be one or more elements that temporarily or permanently store data and instructions. Although memory 920 is illustrated as part of computer 900, memory can also be implemented in network 990, in computers 901/902 and in processor 910 itself (e.g., cache, register), or elsewhere. Memory 920 can be a read only memory (ROM), a random access memory (RAM), or a memory with other access options. Memory 920 is physically implemented by computer-readable media, for example: (a) magnetic media, like a hard disk, a floppy disk, or other magnetic disk, a tape, a cassette tape; (b) optical media, like optical disk (CD-ROM, digital versatile disk - DVD); (c) semiconductor media, like DRAM, SRAM, EPROM, EEPROM, memory stick.

[016] Optionally, memory 920 can be distributed. Portions of memory 920 can be removable or non-removable. For reading from media and for writing in media, computer 900 may use well-known devices such as, for example, disk drives, tape drives, etc.

[017] Memory 920 stores modules such as, for example, a basic input output system (BIOS), an operating system (OS), a program library, a compiler, an interpreter, and a text- processing tool. Such modules are commercially available and can be installed on computer 900. As they are well known in the state of the art, such modules are not illustrated in FIG. 1.

[018] CPP 100 has program instructions and, optionally, data that cause processor 910 to execute method steps consistent with the present invention. In other words, CPP 100 can control the operation of computer 900 and its interaction in network system 999 so that is operates to perform in accordance with the invention. For example and without the intention to be limiting, CPP 100 can be available as source code in any programming language, and as object code ("binary code") in a compiled form.

[019] Although CPP 100 is illustrated as being stored in memory 920, CPP 100 can be located elsewhere. CPP 100 can also be embodied in carrier 970.

[020] Carrier 970 is illustrated outside computer 900. For communicating CPP 100 to computer 900, carrier 970 may be conveniently inserted into input device 940. Carrier 970 may be implemented as any computer readable medium, such as a medium largely explained above (cf. memory 920). Carrier 970 may be an article of manufacture having a computer readable medium with computer readable program code to cause the computer to perform methods of the present invention. Further, signal 980 can also embody computer program product 100.

[021] Examples of CPP 100, carrier 970, and signal 980 have been described in connection with computer 900. However, other type of carriers and signals may embody computer program products (CPP) to be executed by further processors in computers 901 and 902.

[022] Input device 940 provides data and instructions for processing by computer 900. Device 940 can be, for instance, a keyboard, a pointing device (e.g., mouse, trackball, cursor direction keys), microphone, joystick, game pad, scanner, or disc drive. Although the examples are devices with human interaction, device 940 can

also be a device without human interaction, for example, a wireless receiver (e.g., with satellite dish or terrestrial antenna), a sensor (e.g., a thermometer), a counter (e.g., a goods counter in a factory). Input device 940 can serve to read carrier 970.

[023] Output device 950 presents instructions and data that have been processed. For example, this can be a monitor or a display, (cathode ray tube (CRT), flat panel display, liquid crystal display (LCD), speaker, printer, plotter, vibration alert device. Output device 950 can communicate with the user, but it can also communicate with further computers.

[024] Input device 940 and output device 950 can be combined to a single device. Any device 940 and 950 can be provided optionally, as they may or may not pertain to various embodiments of the present invention.

[025] Bus 930 and network 990 provide logical and physical connections by conveying instruction and data signals. While connections inside computer 900 are conveniently referred to as "bus 930", connections between computers 900-902 are referred to as "network 990". Optionally, network 990 includes gateways which are computers that specialize in data transmission and protocol conversion.

[026] Devices 940 and 950 are coupled to computer 900 by bus 930 (as illustrated) or by network 990 (optional). While the signals inside computer 900 are mostly electrical signals, the signals in network may be electrical, electromagnetic, optical or wireless (radio) signals.

[027] Networks are commonplace in offices, enterprise-wide computer networks, intranets and the Internet (e.g., world wide web or WWW). Network 990 can be a wired or a wireless network. To name a few network implementations, network 990 can be, for example, a local area network (LAN), a wide area network (WAN), a

public switched telephone network (PSTN); a Integrated Services Digital Network (ISDN), an infra-red (IR) link, a radio link, like Universal Mobile Telecommunications System (UMTS), Global System for Mobile Communication (GSM), Code Division Multiple Access (CDMA), or satellite link.

[028] A variety of transmission protocols, data formats and conventions is known and may include, for example, transmission control protocol/internet protocol (TCP/IP), hypertext transfer protocol (HTTP), secure HTTP, wireless application protocol (WAP), unique resource locator (URL), a unique resource identifier (URI), hypertext markup language (HTML), extensible markup language (XML), extensible hypertext markup language (XHTML), wireless markup language (WML), and Standard Generalized Markup Language (SGML).

[029] Interfaces coupled between the elements are also well known in the art, and not shown in FIG. 1. An interface can be, for example, a serial port interface, a parallel port interface, a game port, a universal serial bus (USB) interface, an internal or external modem, a video adapter, or a sound card.

[030] Computers and the programs that operate them are closely related. As used hereinafter, phrases, such as "the computer provides" and "the program provides," are used interchangeably to express actions by a computer that is controlled by a program.

[031] FIG. 2 is a diagram illustrating translation code generation after a change in the object model, according to one or more embodiments of the present invention. In FIG. 2, a client 202 and a server 200 are shown. The client 202 comprises a data object definition tool 204a and a data management tool 204b. Server 200 comprises a

translation code generator 206 and a data translation service 208. Depicted is a flow chart of messages between the client 202 and the server 200.

[032] During normal operation of the embodiment shown in FIG. 2, data management tool 204b requests data from data translation service 208 data of the server 200. This is done via a request message Data_Req. Within data translation service 208, the data requested by the client 202 is extracted from the data in the server and put into a data format required by the client 202. The data is translated into a data format for the client 202 using translation code that has been adapted to the data model of the client 202. Thus, the data within the server is put into the right format and then transmitted from the server 200 to the client 202 in a response message Data_Res.

[033] In the event that the data model within the client 202 changes, element A in this embodiment of FIG. 2, the data object definition tool 204a transmits a client data object definition message CDOD to translation code generator 206 within server 200. Upon receipt of this client data object definition message CDOD, translation code generator 206 generates a new translation code, element B, depending on the object model of the client 202. The data object definition message comprises information about the data format requested by the client 202. After the translation code is generated B this translation code is activated and transmitted to data translation service 208.

[034] After activation of a new translation code, this translation code may be used by data translation service 208. In this case, after receipt of a new data request message Data_Req from data management tool 204b, data translation service 208 translates the data in the server according to the new translation code, and responds to the request message by transmitting the data in the correct data format Data_Res.

[035] By adapting the translation code automatically, no manual coding or adaptation of the translation code is necessary.

[036] FIG. 3 is a diagram illustrating translation code generation after receiving data comprising a new data object definition, according to one or more embodiments of the present invention. During normal operation of the embodiment of FIG. 3, data management tool 204 requests data Data_Req from data translation service 208 within server 200. Data translation service 208 translates data within the server and provides the translated data in an appropriate data format to the client 202.

[037] In the event that the data model within the client 202 changes, element A, the data object definition tool 204a informs data management tool 204b about the new data model in a new data model message. Upon the next data request Data_Req, the data request message may comprise information that a new data model is supported by client 202. The receiving server 200, in particular the data translation service 208, detects that a new data model is supported by the client 202. This may be done, for example, by using version management within data translation service 208.

[038] If a new data model is required by the client 202 and detected within the server 202, translation code generator 208 may request CDOD_Req a data object definition message from data object definition tool 204a. Data object definition tool 204a provides the translation code generator 208 with the new client data object definition message in a response message CDOD_Res. In translation code generator 206, a new translation code is generated B.

[039] New translation code B is activated and provided to data translation service 208. Data translation service 208 uses the new translation code for translating the data requested by the client in the previous data request message. The data

response message Data_Res now is in an appropriate format as the new translation code has been used.

[040] Within data translation service 208, or within any other entity of server 200, an authorization control may be activated. Such activation means that the client 202 may only access data and may only initiate a change within the translation code if it has authorization to do so. This authorization may be checked by the server 200.

[041] In certain embodiments, the data object definition message is automatically transmitted from the client to the server upon change of the data format within the client. In this case, a data object definition, e.g. a data format, has been modified at the side of the client. This modification may trigger transmitting the data containing the data object definition message from the client to the server. The server might only need to listen to incoming data object definition messages and may then use these messages for extracting the information requested by the clients and for re-coding automatically the translation code.

[042] In certain embodiments, the translation code is adapted to the changed data format within a translation code generator upon reception of the data object definition message. Within the translation code generator different translation codes may be stored for different clients and different systems.

[043] According to at least one embodiment, the translated data may be transmitted from the server to the client using a standard object description language like, for example an extended markup language (XML). By using a standard object description language, implementation is simplified and programmers may use their common programming knowledge. The same applies for an embodiment wherein the data object definition message is transmitted from the client to the server using a

standard object description language, for example an extended markup language. The data object definition message may then comprise content and structure, as XML allows storing structured content.

[044] According to certain embodiments, the data required by the client is extracted and translated from the stored data by the translation code prior to sending the translated data from the server to the client. The translation code is used for extracting the relevant data from the data stored in the server. The relevant data may then be translated into a data format preferred by the particular client. This may be done prior to sending the data to the client.

[045] By using XSL for translating in the translation code the data into the data format used by the client, standard programming routines and programming knowledge may be used.

[046] In certain embodiments, to avoid "information loops" with respect to the data format of the data provided to the clients, it may be important to fix the roles of server and client within the process. The server must publish and have the ownership of the data definition format. A client may be responsible for definition and distribution of the meta-data of the data objects being used.

[047] Since data object definition messages received within the server may directly modify code on the server side, authorization management may be used in order to control and restrict the reaction of the server to different data object definition messages. Therefore, access to the server by the data object definition messages may be controlled by authorization management. The authorization management may be comprised within the server. It may keep track of clients authorized to initiate translation code generation, which may be so-called "trusted systems". Moreover, authorization

may be based on user “roles”. Each user may have certain rights. E.g. system administrators may have all the same rights. Coded information about the users may be sent embedded in messages exchanged between servers and clients.

[048] To connect more than one data client to a single server, data formats of different clients may be managed by version management within the server. In this case, a particular translation code for a particular client may be stored and a different translation code may be used for each client. Also clients of different groups, e.g. different data management systems, may use one particular translation code. Version identification may be based on two parameters, which may be encapsulated and exchanged within messages between clients and servers. One parameter may comprise information concerning the data model version, e.g. by using time stamps or other string combinations. This allows identification within the system. Another parameter may allow the identification of a system within a computer network. This could be for example an IP number.

[049] In at least one embodiment of a computer-implemented method consistent with the present invention, upon change of the data format, the server requests the data object definition message from the client and the client transmits the data object definition message upon request to the server. By using data format version identification within a server, the server may detect automatically changes in the data format of the client. This may be done during exchange of master data. When a new data format is detected by the server, the server may trigger the request of a new data object definition message. Upon receipt of this data object definition message, the server may trigger a generation of a new data translation code for the given client. The

existing translation code may then be re-coded. Thereafter, data to be transmitted to the client will be translated by the new translation code.

[050] A further aspect of the invention is a computer program product for automatically configuring a translation code, the program comprising instructions operable to cause the processor to store data within a server, translate the data into a data format required by a client using the translation code within the server, transmit the translated data from the server to the client, transmit a change of the data format from the client to the server in a data object definition message and adapt automatically the translation code to the changed data format upon reception of the data object definition message.

[051] Yet a further aspect of the invention is a computer system for automatically configuring a translation code with storage means for storing data within a server, the server comprising a translation code generator, and the server comprising translation means for translating data into a data format required by a client using a translation code from the translation code generator, transmission means for transmitting the translated data from the server to the client, and transmitting a change of the data format within a data object definition message from the client to the server wherein the translation code generator comprising adaptation means for adapting the translation code automatically to the changed data format upon reception of the data object definition message.

[052] While the invention has been particularly shown and described with reference to examples and embodiments described above and illustrated in the figures, it is to be understood by those skilled in the art that various changes in form and detail may be made without departing from the spirit and scope of the invention. For example,

the scope of the invention is in no way limited to the exemplary computer systems described and illustrated herein. In sum, the metes and bounds of the present invention include the described embodiments and examples as well as all variations and modifications thereof, and shall be limited soley by the scope of the claims appended hereto.

REFERENCE NUMBERS

200	server
202	client
204a	data object definition tool
204b	data management tool
206	translation code generator
208	data translation service
900	computer
910	processor
920	memory
930	bus
940	input device
950	output device
960	user interface
970	program carrier
980	program signal
999	computer network system